# Department of Computing

**CS-213: Advanced Programming**

**Class: BSCS 7AB**

# Lab 07: Express JS

**Date: 17 October, 2019**

**Time: 10:00-01:00pm & 02:00-05:00pm**

**Submitted By: Muhammad Nouman Imran (215192)**

# Instructor: Dr. Sidra Sultana

**Lab Engineer: Ms. Ayesha Asif**

# 

# Lab 07: Express JS

**Introduction**

Express is a minimal and flexible Node.js web application framework that provides a robust set of features to develop web and mobile applications. It facilitates the rapid development of Node based Web applications.

**Objectives**

This lab will get you familiar with the Node Express JS environment.

**Tools/Software Requirement**

Node.js, Express Js, Notepad

**Lab Tasks**

**Task 1:** Create Index.html file in the root folder of your application and write the HTML FORM POST method code in it. Modify server.js to handle home page requests as well as the input sent by the HTML form.

**Hint:** To handle HTTP POST request in Express.js version 4 and above, you need to install middleware module called body-parser. The middleware was a part of Express.js earlier but now you have to install it separately. This body-parser module parses the JSON, buffer, string and url encoded data submitted using HTTP POST request. Install body-parser using NPM as shown below.

**\*\* Server.js \*\***

**const express = require('express');**

**const app = express();**

**const path = require('path');**

**const router = express.Router();**

**var bodyParser = require('body-parser');**

**var multer = require('multer');**

**var upload = multer();**

**router.get('/',function(req,res){**

**res.sendFile(path.join(\_\_dirname+'/index.html'));**

**//\_\_dirname : It will resolve to your project folder.**

**});**

**app.use(bodyParser.json());**

**app.use(bodyParser.urlencoded({ extended: true }));**

**//form-urlencoded**

**// for parsing multipart/form-data**

**app.use(upload.array());**

**app.use(express.static('public'));**

**app.post('/', function(req, res){**

**console.log(req.body);**

**res.send("recieved your request!");**

**});**

**//add the router**

**app.use('/', router);**

**app.listen(process.env.port || 3000);**

**console.log('Running at Port 3000');**

**\*\* Index.html \*\***

**<html>**

**<head>**

**<title>Express HTML</title>**

**</head>**

**<body>**

**<form method="POST" action="/">**

**First Name: <input type="text" name="fname">**

**<br>**

**Last Name: <input type="text" name="lname">**

**<br>**

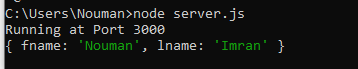
**<button type="Submit">Submit</button>**

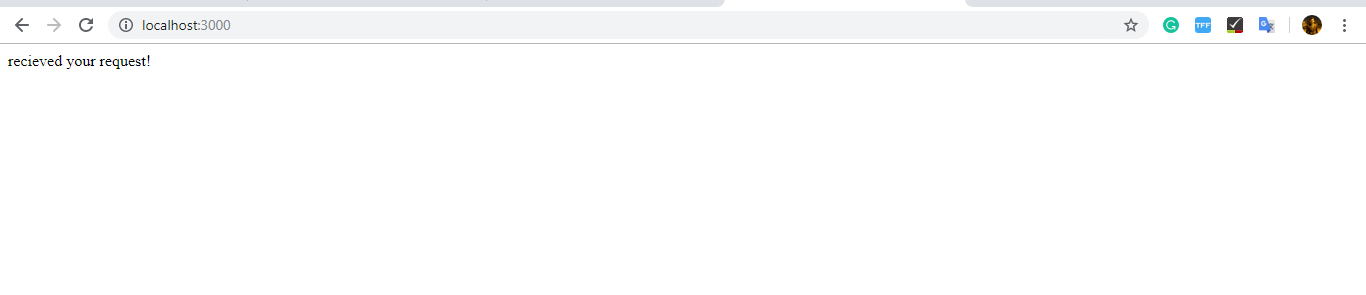
**</form>**

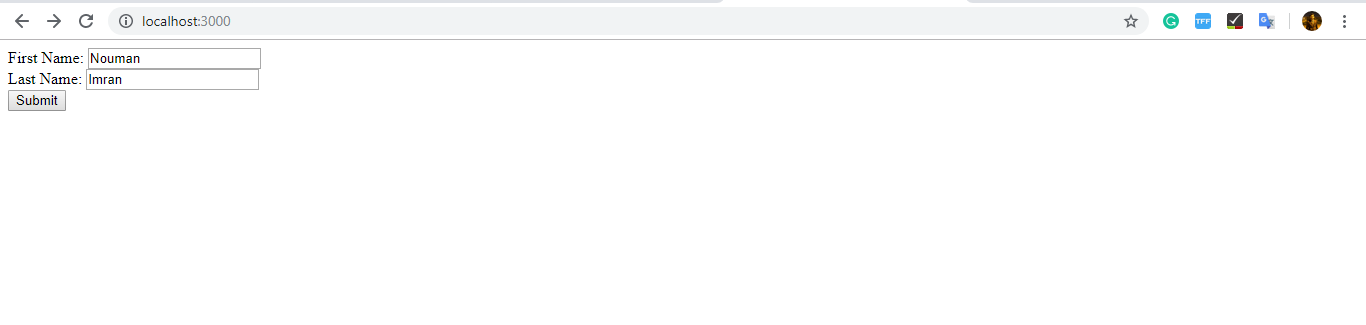
**</body>**

**</html>**

**Output:**







**Task 2:** Create a file uploader form in an html file. This form has method attribute set to POST and enctype attribute is set to multipart/form-data. Modify server.js to handle home page requests as well as file upload.

**\*\* expressfile.js \*\***

**Note: UPLOAD FOLDER MUST EXIST IN THE UERS IN C DIRECTORY WHERE THE FILES WILL BE UPLOADED**

**OTHERWISE IT WILL GIVE ERROR, AND THE FILE IS SAVE AS FILE FORMAT SO MUST BE OPENED WITH THE RELEVANT**

**SOTFWRAE....**

**// call all the required packages**

**const express = require('express')**

**const bodyParser= require('body-parser')**

**const multer = require('multer');**

**//CREATE EXPRESS APP**

**const app = express();**

**app.use(bodyParser.urlencoded({extended: true}))**

**//ROUTES WILL GO HERE**

**app.get('/',function(req,res){**

**res.sendFile(\_\_dirname + '/index.html');**

**});**

**//server.js**

**// SET STORAGE**

**var storage = multer.diskStorage({**

**destination: function (req, file, cb) {**

**cb(null, 'uploads')**

**/\*NOTE: UPLOAD FOLDER MUST EXIST IN THE UERS IN C DIRECTORY WHERE THE FILES WILL BE UPLOADED**

**OTHERWISE IT WILL GIVE ERROR, AND THE FILE IS SAVE AS FILE FORMAT SO MUST BE OPENED WITH THE RELEVANT**

**SOTFWRAE.....**

**\*/**

**},**

**filename: function (req, file, cb) {**

**cb(null, file.fieldname + '-' + Date.now())**

**}**

**})**

**var upload = multer({ storage: storage })**

**//upload txt files**

**app.post('/uploadfile', upload.single('myFile'), (req, res, next) => {**

**const file = req.file**

**if (!file) {**

**const error = new Error('Please upload a file')**

**error.httpStatusCode = 400**

**return next(error)**

**}**

**res.send(file)**

**})**

**//upload images**

**app.post('/upload/photo', upload.single('myImage'), (req, res, next) => {**

**const file = req.file**

**if (!file) {**

**const error = new Error('Please upload a file')**

**error.httpStatusCode = 400**

**return next(error)**

**}**

**res.send(file)**

**})**

**app.listen(3000, () => console.log('Server started on port 3000'));**

**\*\* Index.html \*\***

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<title>MY APP</title>**

**</head>**

**<body>**

**<!-- SINGLE FILE -->**

**<form action="/uploadfile" enctype="multipart/form-data" method="POST">**

**<input type="file" name="myFile" />**

**<input type="submit" value="Upload a file"/>**

**</form>**

**</form>**

**<form action="/upload/photo" enctype="multipart/form-data" method="POST">**

**<input type="file" name="myImage" accept="image/\*" />**

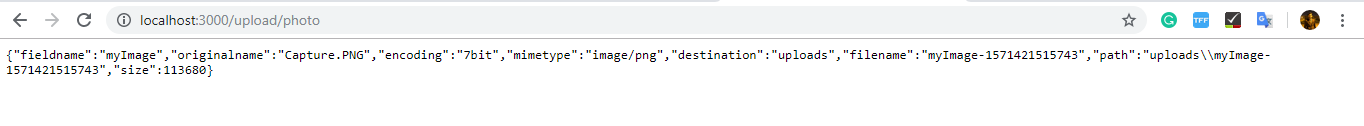
**<input type="submit" value="Upload Photo"/>**

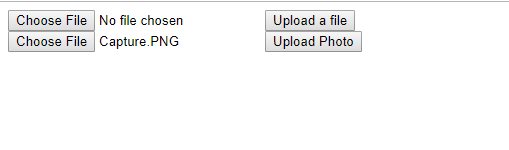
**</form>**

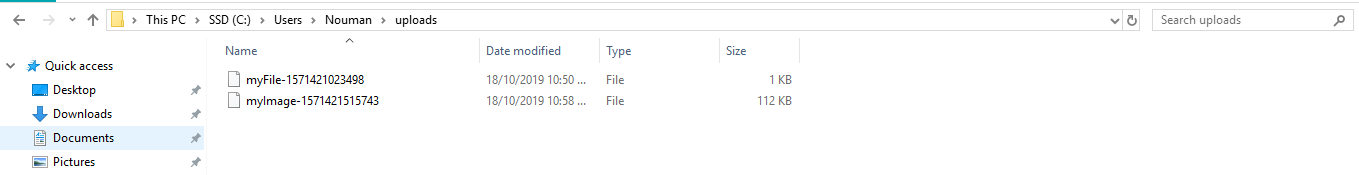
**</body>**

**</html>**

**Output:**







**Task 3:** By using the Pug templating engine create an HTML registration form.

**\*\* server.js file \*\***

**var express = require('express');**

**var bodyParser = require('body-parser');**

**var multer = require('multer');**

**var upload = multer();**

**var app = express();**

**app.get('/', function(req, res){**

**res.render('form'); });**

**app.set('view engine', 'pug');**

**app.set('views','./views');**

**app.use(bodyParser.json()); // for parsing application/xwww- app.use(bodyParser.urlencoded({ extended: true })); //form-urlencoded**

**//for parsing multipart/form-data**

**app.use(upload.array());**

**app.use(express.static('public'));**

**app.post('/', function(req, res){**

**console.log(req.body);**

**res.send("recieved your request!");**

**}); app.listen(3000);**

**\*\* PUG Template: \*\***

**html**

**head**

**HTML Form in Pug**

**body**

**center**

**h1 HTML Form in html**

**form(method='POST')**

**input(name='fname', placeholder='Enter name...')**

**br**

**br**

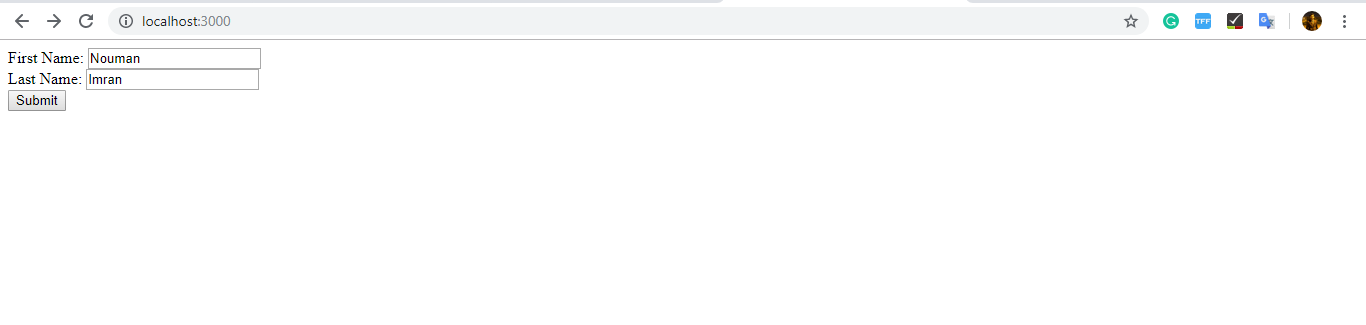
**input(type='number', name='age', placeholder='Enter age...')**

**br**

**br**

**button(type='submit') Submit**

**Output:**



### Deliverables

Compile a single word document by filling in the solution part and submit this Word file on LMS. This lab grading policy is as follows: The lab is graded between 0 to 10 marks. The submitted solution can get a maximum of 5 marks. At the end of each lab or in the next lab, there will be a viva/quiz related to the tasks. You must show the implementation of the tasks in the designing tool, along with your complete Word document to get your work graded. You must also submit this Word document on the LMS. In case of any problems with submissions on LMS, submit your Lab assignments by emailing it to Ms. Ayesha Asif: [ayesha.asif@seecs.edu.pk](mailto:ayesha.asif@seecs.edu.pk).